Volume 0 (1981), Number 0 pp. 1–12 COMPUTER GRAPHICS *forum*

Instant Caching for Interactive Global Illumination

Kurt Debattista[1], Piotr Dubla[1], Francesco Banterle[1], Luís Paulo Santos[2], and Alan Chalmers[1]

[1] WMG, International Digital Laboratory, University of Warwick, United Kingdom
[2]Departamento de Informatica, Universidade do Minho, Portugal

Abstract

The ability to interactively render dynamic scenes with global illumination is one of the main challenges in computer graphics. The improvement in performance of interactive ray tracing brought about by significant advances in hardware and careful exploitation of coherence has rendered the potential of interactive global illumination a reality. However, the simulation of complex light transport phenomena, such as diffuse interreflections, is still quite costly to compute in real time. In this paper we present a caching scheme, termed Instant Caching, based on a combination of irradiance caching and instant radiosity. By reutilising calculations from neighbouring computations this results in a speedup over previous instant radiosity-based approaches. Additionally, temporal coherence is exploited by identifying which computations have been invalidated due to geometric transformations and updating only those paths. The exploitation of spatial and temporal coherence allows us to achieve superior frame rates for interactive global illumination within dynamic scenes, without any precomputation or quality loss when compared to previous methods; handling of lighting and material changes is also demonstrated.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism I.3.3 [Computer Graphics]: Picture/Image Generation

1. Introduction

Achieving global illumination in real time remains one of the major goals in the field of rendering. Unfortunately, interactive global illumination is a computationally complex task. Recent improvements in the field of ray tracing have made it possible to achieve Whitted-style ray tracing in real time for dynamic scenes on desktop PCs [WMG*07]. This allows for the interactive computation of many of the global illumination effects, such as specular phenomena and correct shadows by recalculating them at each frame; other effects, such as diffuse interreflections, require simulations of the light transport that make it hard to maintain real time frame rates.

Diffuse interreflections are commonly accelerated by exploiting the property that indirect diffuse light changes smoothly across surfaces. These approaches are based on precomputation [GTGB84, SKS02], on caching using an on-demand view-driven evaluation of interreflections for re-use across space [WRC88] or on shooting luminous flux from light sources at a first stage and reutilising this information for the rendering pass [Jen01]. Although suitable for offline systems, these techniques are not entirely appropriate for real time rendering, since they entail costly operations, such as transport precomputation, radiance gathering or density estimation.

Dynamic scenes, with changing geometry, lighting and materials, represent a further challenge for interactive global illumination. Precomputed and caching methods are seldom used in this context because changes in the scene invalidate existing information; correctly updating the cached data is both complex and prone to temporal flickering artifacts. Consequently, certain methods, such as Instant Global Illumination (IGI) [WKB*02], recompute all light paths for every frame, thus ignoring any potential temporal coherence.

This paper proposes a new interactive caching scheme for indirect diffuse interreflections within dynamic scenes based on exploiting spatial and temporal coherence enabling interactive global illumination on a single multicore PC. The method, referred to as Instant Caching, is based on computing illumination from virtual point light sources (VPLs), shot in a similar way to instant radiosity [Kel97], and caching the VPLs' contributions for spatial reutilisation across the same

Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

(a) (b) (c)

Figure 1: *Instant caching being used to interactively render a Cornell box (ca. 65K triangles) with indirect diffuse interreflections within a ray-tracing framework at around 10fps. Our method allows us to interactively change the view (a), select and move objects (b), change materials and move light sources (c).*

frame, thus exploiting spatial coherence similarly to the irradiance cache [WRC88]. The combination of instant radiosity and irradiance caching improves rendering time upon the former by interpolating over object space and achieves significant speedup upon the latter by having multiple light bounces handled by the VPL shooting stage.

Instant Caching naturally adapts to dynamic geometry within an interactive application by exploiting temporal coherence among frames. VPLs are shot using a quasi-random distribution to ensure maximum temporal coherence; cached indirect diffuse light paths invalidated by geometric transformations are identified at each frame and only these need be recomputed. This approach is referred to as Temporal Instant Caching.

The main contribution of this paper to interactive global illumination is thus the identification and update of all invalid indirect diffuse light paths resulting from geometric transformations. The algorithm reduces temporal noise and computational workload (as compared to brute force approaches) by exploiting temporal coherence, thus resulting in an improved frame rate; these results are achieved without the need for any additional data structures or any knowledge of the animation paths. Lighting and material changes are also accommodated within the same approach.

2. Related Work

Many interactive rendering techniques for global illumination have focused on sparse sampling methods, which compute results for certain portions of the image and re-use the computation across time and space. The render cache [WDP99] used image-space methods to re-use computation, while the shading cache [TPWG02] computed global illumination at the vertices of an adaptive mesh and performed interpolation on object space. The work by Bala et al. [BWG03] used an edge-and-point map to ensure that the interpolation did not occur across discontinuities of geometry and shadows. The concept was further used for caching samples across frames within a frameless rendering framework [DWWL05]. These sparse sampling methods were primarily dedicated to achieving interactivity and could display illumination which would not be technically correct, usually visualised as temporal or spatial artifacts.

Interactive rendering solutions that harness GPUs' processing power have recently begun to emerge. PRT [SKS02] used precomputation of the light transport to render static scenes interactively. Recent extensions of PRT [PWL*07, IDYN07], based on initial work [ZHL*05] where the radiance field around each single object was precomputed, have made it possible to interactively render rigid dynamic scenes in real time. Dachsbacher et al. presented an interactive solution in which the rendering equation was reformulated [DSDD07]. This new reformulation makes the visibility implicit, using anti-radiance to compensate for extraneous transport. The solution was evaluated hierarchically on a GPU achieving interactive frame rates for fully dynamic scenes.

Imperfect Shadow Maps allow interactive computation on the GPU of diffuse and glossy indirect illumination in fully dynamic scenes by combining instant radiosity with approximate visibility queries [RGK*08]. By realising that accurate visibility information is not required for glossy and diffuse indirect illumination, the authors compute low resolution shadow maps for each VPL from a crudely simplified point representation of the scene. This method differs from our approach since it does not evaluate correct visibility information and it does not exploit temporal coherence in order to reduce the workload. However, the observation that accurate visibility is not required for diffuse indirect illumination can eventually be exploited to speed up temporal instant caching.

By taking advantage of the processing power of modern CPUs, it has become possible to ray trace scenes interactively. The work of Parker et al. [PMS*99] focused on interactive ray tracing on large shared memory computers. Wald et al. [WSBW01] exploited cache coherence and fast SIMD

instructions to ray trace scenes in real time on clusters of commodity PCs. Recent advances have made it possible to ray trace dynamic scenes in real time, see [WMG*07] for an overview.

Instant radiosity, introduced by Keller [Kel97] and used frequently for interactive rendering, utilised virtual point lights (VPLs) to account for indirect diffuse light paths. VPLs were deposited at intersection points of paths fired from the light sources; their contribution was then calculated via rasterisation using an accumulation buffer. Wald et al. [WKB*02] presented an extension to their fast ray tracing, termed instant global illumination (IGI), by adapting instant radiosity for ray tracing. In IGI, VPLs are shot from the light sources in an initial pass at the beginning of each frame. When rendering, each pixel computes visibility with a subset of the VPLs using interleaved sampling. The resulting structured noise is removed by filtering on image space via a discontinuity buffer. Although IGI explores spatial coherence in image space, it does not exploit temporal coherence, resorting to a brute force approach where all light paths are recomputed for each frame. Laine et al. presented an incremental rendering method for instant radiosity on the GPU which achieved interactive frame rates but did not take into account the contribution of the dynamic objects [LSK*07].

Lightcuts is a scalable algorithm for computing illumination from many point lights with a rendering time strongly sublinear with the number of such lights [WFA*05]. A binary tree and perceptual metric are used to hierarchically partition the lights into clusters; the appropriate cut within this tree is then selected to illuminate each shading point, drastically reducing the number of visibility evaluations with the point lights. The approach was further extended [WABG06] to unify handling of effects such as motion blur, depth of field, spatial anti-aliasing and participating media. Even though temporal coherence has not been investigated by the authors, lightcuts are a natural fit for instant radiosity-based methods. If the lightcuts algorithm can be extended with some form of temporal awareness, such that the VPL clustering does not change drastically from frame to frame, then it may constitute an important extension to temporal instant caching, reducing the VPL processing count.

2.1. Caching schemes

The irradiance cache presented by Ward et al. [WRC88] is a structure that accelerates the computation of indirect diffuse interreflections in a view-driven fashion. Whenever an indirect computation is required the irradiance cache is consulted and if any samples in the irradiance cache are within a search radius they may be extrapolated/interpolated from, to produce the next value. If an appropriate sample is not found, the full indirect diffuse computation is performed and stored for future computation. The use of irradiance gradients [WH92] improved the performance by storing translational and rotational gradients with the cached samples. Tabellion and Lamorlette [TL04] adopted irradiance caching within the framework of computer generated imagery for entertainment. Arikan et al. [AFO05] improved caching performance by decoupling the final gathering of distant geometry and local geometry. Krivanek et al. [KGPB05] presented the radiance cache, a method that could cache glossy interreflections in addition to diffuse interreflections. Krivanek et al. [KBPv06] further improved the quality and performance of (ir)radiance caching by adapting the search radius during computation. Caching methods have been extended for use with other illumination effects such as subsurface scattering [KLC06] and participating media [JDZJ08]. Gautron et al. [GKBP05] presented an (ir)radiance cache implementation on the GPU; while faster than traditional irradiance caching, only one bounce indirect diffuse light transport is simulated.

For all the above mentioned approaches the cached computations could be re-used by subsequent frames as long as the scene remains static. Of particular relevance to our work are the techniques based on (ir)radiance caching that make some attempt at temporal caching of samples for re-use in dynamic scenes. As with the caching methods, none of these methods have been used for real time interactive environments. Tawara et al. [TMS04] extended the irradiance cache into the temporal domain by updating the irradiance cache sample's visibility rays, used in the final gathering part of photon mapping, over time. Their ageing scheme identified the oldest samples and updated them. This method will typically use invalid samples at any given time when an object is moving.

Smyk et al. [SKDM05] presented a temporally coherent irradiance caching method based on caching the final gathering rays in photon mapping and a new anchor data structure. Anchors were used to group final gathering rays (constituting the computation of the cached sample) with the photons. The final gathering rays were linked with the closest anchor and when unavailable a new anchor was created on demand. Changes in the irradiance value at an anchor would trigger updating the irradiance cache strata linked with it. This method requires the use of an extra abstract data structure (the anchor kd-tree) to bind the photons with the final gathering rays and all the mechanisms related with it to manage anchor creation, update and deletion, on top of photon mapping and irradiance caching. Our algorithm does not require such an extra data structure since each sampled direction in every irradiance cache record is implicitly linked to a VPL, thus dispensing additional data structures. Gautron et al. [GBP07] presented a temporal (ir)radiance cache that catered for changes in the temporal domain and included computation of a temporal gradient. Their method predicted incoming lighting and how it would affect the cached samples. Animation paths needed to be known beforehand to correctly predict illumination.

3. Instant Caching

Our instant caching method caches the irradiance from VPLs as opposed to using hemispherical sampling, as is the case with irradiance caching. This caching scheme serves a dual purpose accounting for increased performance in the spatial and temporal domains. Firstly, in the spatial domain, it is used to accelerate the computation of each frame by interpolating over object space, following an approach similar to the irradiance cache. Instant caching, however, requires shooting a single ray, referred to as a visibility ray, to evaluate each VPL contribution, multiple light bounces being handled by the VPLs shooting stage. The irradiance cache requires either initiating a tree of rays or originating a random walk to evaluate each stratum contribution; either way, the computation is more than that of an instant cache visibility ray. Some methods have used irradiance caching in conjunction with photon mapping, thus not incurring the multiple bounces described above. However, the cost of extra visibility tests is replaced by photon shooting and density estimation. Additionally, the visibility test in instant caching may be more memory coherent since visibility rays are traced towards the same VPLs. Secondly, in the temporal domain, updating each VPL contribution to a given cache sample due to scene transformations requires a single reevaluation of the respective visibility ray. Often, given the proposed optimisations (see Section 3.2), the visibility ray is intersected with only a subset of the geometry, improving interactive performance due to reutilisation of information from previous frames.

3.1. Static Instant Caching

The static algorithm, applied to the first frame of an animation, entails two steps: shooting photons from light sources and creating VPLs, in a manner similar to instant radiosity and IGI [Kel97, WKB*02], and a subsequent gathering pass. The gathering pass evaluates indirect diffuse irradiance incoming from VPLs at a sparse set of points and interpolates among them for the remaining ones, similarly to the irradiance cache.

The evaluation of the indirect diffuse component using instant radiosity at a point x requires the calculation of the contribution of each VPL, which equates to:

$$L_{indirect}(\mathbf{x},\omega_o) = \sum_{k=1}^{N} f_r(\mathbf{x},\omega_o,\omega_k) L_{e,k} V(\mathbf{x},\mathbf{y}_k) G'(\mathbf{x},\mathbf{y}_k), \tag{1}$$

where N is the number of VPLs in the scene, $V(\cdot,\cdot)$ is the visibility function between two points, ω_k is the light vector for the k-th VPL, $L_{e,k}$ is the emitted radiance of the k-th VPL, $\mathbf{y}_k$ is the position of the k-th VPL, $f_r(\mathbf{x},\omega_o,\omega_k) = \rho/\pi$ in the case of diffuse component (ρ is the reflectance), and G' is the bounded geometry term [PH04]. The bounded geometry term is defined as:

$$G'(\mathbf{x},\mathbf{y}) = \frac{\cos^+\theta_{\mathbf{x}} \cos^+\theta_{\mathbf{y}}}{\|\mathbf{x}-\mathbf{y}\|_2^2} f(0.8 min_d, 1.2 min_d, \|\mathbf{x}-\mathbf{y}\|_2), \tag{2}$$

where $\cos^+\theta_{\mathbf{x}}$ is the cosine of the angle between the light vector ω_k and normal at $\mathbf{x}$, $\cos^+\theta_{\mathbf{y}}$ is the cosine of the angle between the ω_k and the normal at $\mathbf{y}_k$, min_d is the bounding distance, and f is the smoothing function:

$$f(a,b,x) = \begin{cases} 1 & \text{if } x > b \\ 3\left(\frac{x-a}{b-a}\right)^2 - 2\left(\frac{x-a}{b-a}\right)^3 & \text{if } a \le x \le b \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Equation 1 is quite expensive to evaluate. To speed-up this calculation, caching, similar to [WRC88], can be employed, where a set of cached samples, Ω, are used for interpolation:

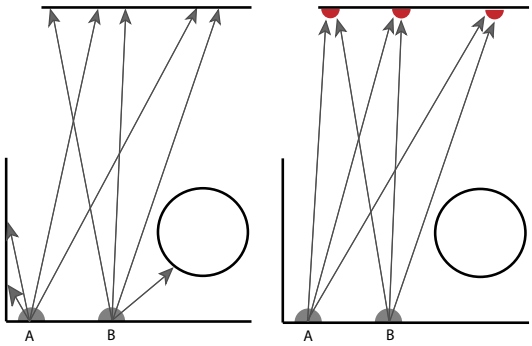
$$L_{indirect}(\mathbf{x},\omega_o) \approx \frac{\rho}{\pi} \sum_{k \in S(\mathbf{x})} \frac{E_k w_k(\mathbf{x})}{\sum_{k \in S(\mathbf{x})} w_k(\mathbf{x})}, \tag{4}$$

where $S(\mathbf{x}) = \{k | k \in \Omega \wedge w_k(\mathbf{x}) > 1/a\}$, a is the caching radius, E_k is the cached irradiance for the k-th sample in the cache, and w_k is the weight for the k-th cached sample used which is calculated as:

$$w_k(\mathbf{x}) = \left(\|\mathbf{x}-\mathbf{x}_k\| + \sqrt{1-(\mathbf{n}\cdot\mathbf{n}_k)}\right)^{-1}, \tag{5}$$

where $\mathbf{n}$ is the normal at $\mathbf{x}$, $\mathbf{x}_k$ and $\mathbf{n}_k$ are respectively the position and normal of the k-th sample. This metric differs from the original irradiance cache one [WRC88], because the harmonic mean distance has been removed from the calculation of $w_k(\mathbf{x})$. The harmonic mean distance, when used with hemisphere sampling, is a value that gives a measure of the density of the geometry surrounding a given point. When using VPLs the visibility rays will seldom be well distributed over the hemisphere, thus the harmonic mean of the rays' length does not necessarily give an indication of the surrounding geometry density. This can be illustrated in the case of a point close to a corner, the computation of the harmonic mean distance with the well-distributed VPLs would not necessarily indicate the proximity to a corner. This rationale is illustrated in Figure 2. Our assumption was further confirmed by the lack of any perceivable differences in quality between images computed with and without a harmonic mean. We have run HDR-VDP comparisons [MDMS05] (HDR-VDP is a perceptual metric; see Section 4.3 for a detailed explanation) among images generated with and without harmonic mean for all the scenes shown in Figure 4 and

the greatest difference was 0.1% and for most images the result was 0.0%.



(a) Traditional Irradiance Cache (b) Instant Cache

Figure 2: *A comparison between the irradiance cache and the instant cache. In the irradiance cache, the harmonic mean distance for the cached sample A will be different from that in B, possibly resulting in further samples near the corner. In the instant cache, the harmonic mean distance would be independent from the geometry context but dependent on the VPLs distribution; in this particular example the harmonic mean of A and B is similar.*

3.2. Temporal Instant Caching

To maintain the correctness of our instant cache samples we need to account for when objects move and invalidate the instant caching samples accordingly. The issues that affect the fidelity of our cache samples occur when an object occludes the path of a visibility ray or VPL, or when an object moves away from the visibility ray or VPL, commonly called deocclusion. Finally, when cached samples lie on dynamic objects the computation may be invalid. We summarise these possible situations into five cases as shown in Figure 3. Case 1 and Case 2 demonstrate the case when VPLs are occluded or deoccluded by moving objects. Case 3 and Case 4 demonstrate the cases when the visibility rays are occluded or deoccluded by dynamic objects. Case 5 demonstrates the case when the cache sample lies on a dynamic object.

Our instant cache maintains all the cached samples within an octree representation. For each cached sample we store position, orientation, irradiance and an array for each VPL representing the contribution to each of the visibility rays. While this may seem like a large amount of memory, typically 256 VPLs are sufficient. Furthermore, a format similar to RGBE [War91] can be employed for storage, resulting in four bytes per visibility ray.

In order to maintain the strong coupling between the visibility rays and VPLs, the VPLs are traced at the beginning of every frame using Quasi-Monte Carlo sampling. This ensures that the samples are always valid when nothing moves and coherence is maintained. It also makes it easy to identify which VPLs have been affected by moving objects. A simple comparison between new and old VPLs is enough to invalidate the visibility ray corresponding to those VPLs for each of the cached samples. This applies also for multiple VPL bounces. This handles both Case 1 and Case 2. The VPLs shot from light sources that have moved are naturally invalidated in this way.

Case 3 can be identified by a visibility test of the moved objects with the cached samples. While projection onto each cached sample could be used to identify objects that have moved, for simplicity and to avoid GPU access, we shoot the visibility rays from the cached samples towards the corresponding VPLs using ray tracing. We test only against the dynamic objects. This process could be computed against each of the objects individually or an acceleration data structure could be built just for the dynamic objects. If the VPL is occluded then its contribution is set to zero.

For Case 4, deocclusion, a more sophisticated approach is required. Since our method allows any object in the scene to be moved, deocclusion would potentially test against all objects in the scene. However, we use a number of optimisations to minimise the number of times we have to test for this case. Deocclusion only affects visibility rays which were previously occluded (in shadow). Our first test selects those visibility rays for which the contribution on the previous frame was zero. For these, a further test with the previous position of the dynamic objects identifies those visibility rays that may need updating: if a ray does not hit any of the dynamic objects at their previous positions, then it must have struck a static object and no updating is required. The remaining visibility rays may potentially hit a VPL or may be occluded by other geometry. The visibility ray is then tested against the entire scene to ensure it is not occluded by a static object. If no occlusion occurs the new value for this visibility ray is computed. The use of these multiple tests has a further advantage. They make it possible to test the first deocclusion against the old position of the dynamic object's bounding box. While this test is more conservative, it is ideal for when the object's vertices are changing, such as with deformable objects. A further optimisation that reduces the number of objects to be tested occurs if Case 3 is run immediately before Case 4. If a visibility ray is occluded then Case 4 need not be tested for that visibility ray. Finally, we deal with Case 5 by identifying which cached samples were within the bounds of the dynamic objects in the previous frame and remove them.

When identified, invalidated visibility rays can either be reevaluated immediately or flagged. Reevaluation requires removing the previous visibility ray's contribution to the cached sample and adding the new one. On the other hand, flagged cached samples can be updated on demand the moment an interpolation is required from them. Object space areas that are without cached samples in a new frame, because

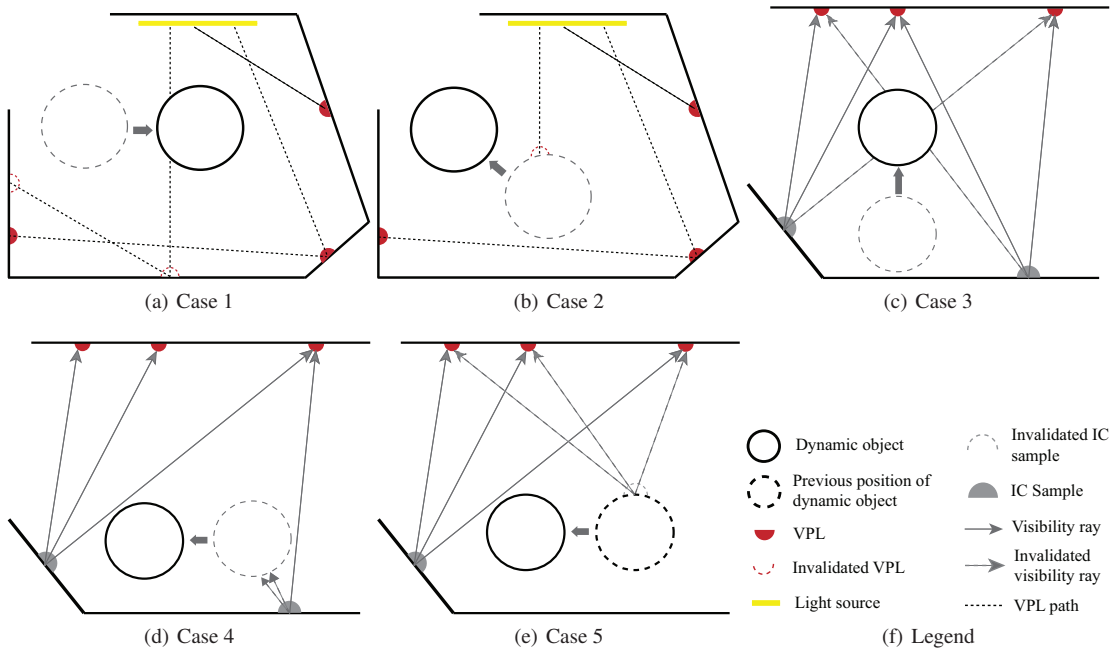


Figure 3: *The five cases that invalidate the instant cache.*

of such areas becoming visible or deletion of cache samples (due to Case 5) are naturally recomputed on demand by the algorithm in the traditional way.

Our method does not need to know beforehand which objects are dynamic and which are not. The dynamic objects need only be identified in the frame in which they are moved. The only aspect that needs to be stored is their position in the previous frame. This feature makes our method well suited for interaction since any object could be moved at any time. As the number of dynamic objects increases, VPLs end up being created at different locations every frame, requiring tracing the corresponding visibility rays from each cache sample against the whole geometry. Additionally, even for those VPLs whose location does not change, the corresponding visibility rays have to be tested against an increasing number of dynamic geometric primitives, eventually approaching the totality of the scene. The performance of the algorithm degrades to that of the static instant cache.

Algorithm 1 demonstrates how we account for all our tests in a single algorithm, in the case of immediate reevaluation, run at the beginning of each frame. Tracing rays against the entire scene is required when a given VPL is invalidated and for special cases as part of the Case 4 test. Case 1 and Case 2 are detected on Line 1 and updated for each VPL from Line 8. Case 5 is handled on Line 3. For Case 3 and Case 4, the tests can be combined into a block test, see Line 11 to Line 21. In this case we can apply the optimisation mentioned for Case 4 above when Case 3 is tested before Case 4. The order of the operations in the algorithm attempts to reduce the cost of the deocclusion operations which are more expensive. In particular the costly testing of the visibility ray against the entire scene in Line 18, is left as the final test, only executed when a number of conditions are not met, improving the overall performance.

Changes on the materials' BRDFs are naturally handled by the algorithm. During the VPLs shooting stage a different BRDF will result in a different spatial distribution or in a different radiant flux for some of the VPLs. VPLs might thus be placed on different locations in the scene and/or their radiant flux might change. Both cases imply invalidation of the respective VPLs from the previous frame and are thus handled in the same manner as Case 1 or Case 2. Material changes on objects where cached samples lie do not require any updating since we cache irradiance, not reflected or transmitted radiance. However, changes of non-Lambertian properties on objects directly visible from the eye are not handled by the algorithm, since this is currently used to shade diffuse surfaces only.

4. Results

In order to demonstrate the effectiveness of instant caching we present some comparative results. Besides implementing the instant caching algorithm we have also implemented IGI [WKB*02]. The acceleration structure used by our ray tracing kernel is a BVH implementation based on Wald et al.'s method [WBS07]. None of our implementations makes

```
Input: Instant Cache: IC, The VPLs: VPLs; The entire scene: Scene, The dynamics objects that have moved: dObjs
/* Function to trace ray across objs returning colour and a hit point hit                    */
Function: hit = Trace(ray, objs, colour);
/* Function to recalculate the visibility ray indexed by index for ICsample based on colour
   */
Function: ReEvaluate(ICsample, index, colour);
1  InvalidateVPLs(); // Case 1 and Case 2
2  foreach ICsample in IC do
3      if ICsample.pos within bounds of previous position of dObjs then
4          Invalidate ICsample and remove from IC; // Case 5
5      else
6          foreach visibility ray in ICsample do
7              if VPLs[VPLindex] invalidated then // Case 1 and Case 2
8                  Trace(visibility ray, Scene, Col);
9                  ReEvaluate(ICsample, VPLindex, Col);
10             else// Case 3 and Case 4
11                 hit = Trace(visibility ray, dObjs, Col);
12                 if hit then // Case 3
13                     ReEvaluate(ICsample, VPLindex, 0.0);
14                 else// Case 4
15                     if old visibility ray was in shadow then
16                         hitOld = Trace(visibility ray, previous position of dObjs, Col);
17                         if hitOld then
18                             if !Trace(visibility ray, Scene, Col) then ReEvaluate(ICsample, VPLindex, Col);
19                         end
20                     end
21                 end
22             end
23         end
24     end
25 end
```

Algorithm 1: *Integrating the invalidation (and possible update) into a single algorithm.*

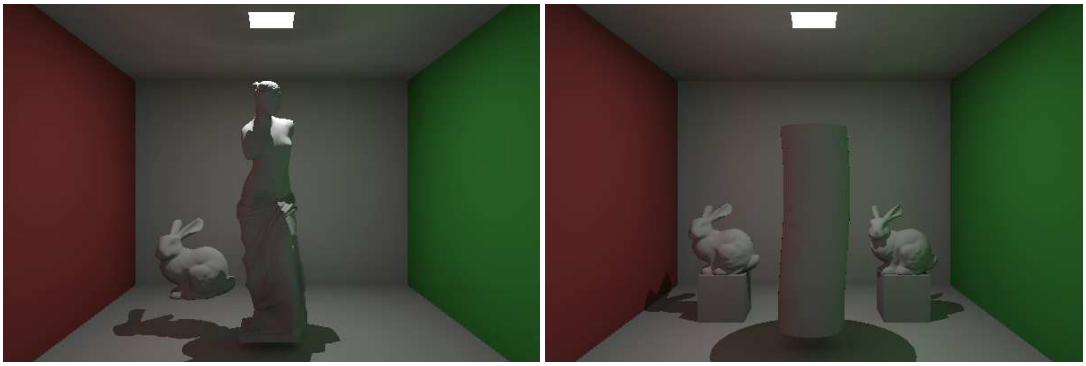
use of packetisation or SIMD operations except for the ray-bounding volume intersection within the BVH traversal [WBS07]. We include the results achieved when using Whitted-style ray tracing to give an indication of our ray tracing performance. For Whitted-style ray tracing we do not compute diffuse interreflections but only specular ones and direct lighting. All images are rendered at a resolution of 600×400 with no super/sub-sampling. Direct lighting is computed for hard shadows only (one shadow ray per light source). IGI and instant caching shoot 256 VPLs. No form of precomputation was used in any of our timings. All results were generated using the scenes shown in Figure 4 on an eight core (dual socket quad-core at 3GHz) with 2GB memory running Mac OS X.

Scenes	**RT**	**IGI**	**IC**	Speedup
Cornell Box	0.05	0.59	0.21	2.81
Wobble	0.06	0.62	0.19	3.26
Desk	0.07	0.76	0.21	3.62
Bunnies	0.08	0.72	0.33	2.18
Cornell Indirect	0.06	0.77	0.24	3.21
Conference	0.14	1.58	0.45	3.51
Sibenik	0.19	1.27	0.85	1.49
Office	0.09	0.78	0.23	3.39

Table 1: *Results, in seconds, for rendering the first frame. Each image was rendered from scratch without relying on any temporal coherence. Speedup compares IC vs. IGI.*

4.1. Static images

Table 1 presents results for the computation of a single image, in order to demonstrate the performance of the static instant cache. Results include rendering times for Whitted ray tracing (RT), IGI and instant caching (IC). Instant caching is the fastest of the tested implementations, achieving a speedup between 1.49 to 3.62 relative to IGI, even though the latter exploits image space coherence by using only a subset of the VPLs per pixel. The improvement in performance is more obvious when viewing the results of diffuse interreflections only; Table 2, shows the timings for these computations and for the same images. In this case for most scenes the speedup is greater than 5. This improvement is due to the exploitation of object space coherence via the in-



(a) Cornell Box (b) Wobble

(c) Desk (d) Bunnies

(e) Cornell Indirect (f) Conference

(g) Sibenik (h) Office

Figure 4: *The scenes used for all experiments.*

Scenes	**IGI**	**IC**	Speedup
Cornell Box	0.43	0.08	5.38
Wobble	0.37	0.05	7.40
Desk	0.56	0.06	9.33
Bunnies	0.57	0.11	5.18
Cornell Indirect	0.55	0.08	6.87
Conference	1.23	0.15	8.27
Sibenik	0.97	0.38	2.55
Office	0.51	0.05	10.02

Table 2: *Results, in seconds, for the diffuse interreflections only when rendering the first frame. Each image was rendered from scratch without relying on any temporal coherence. Speedup compares IC vs. IGI.*

Animation	**RT**	**IGI**	**IC**	**TIC**	Speedup
Cornell Box	17.07	2.11	6.01	10.45	4.95
Wobble	15.84	1.63	2.79	8.95	5.49
Desk	12.33	1.31	4.92	8.94	6.82
Bunnies 1	13.02	1.49	3.42	9.13	6.13
Bunnies 2	12.96	1.45	3.55	8.04	5.54
Bunnies 4	12.81	1.46	3.51	5.81	3.98
Bunnies 9	12.69	1.47	3.54	3.56	2.42
Conference	6.96	1.54	3.37	7.78	5.05
Sibenik	8.23	0.70	1.25	2.63	3.76

Table 3: *Results, in frames per seconds, for rendering the animations. Speedup compares TIC vs. IGI.*

terpolation and is expectable in approaches based on caching and interpolation. The exploitation of coherence will further benefit instant caching when used in the temporal domain (see next section).

4.2. Animations

We present results for temporal instant caching with a number of animations. The animations are comprised of different objects moving around the scenes. For the Cornell Box, the bunny is rotated around the Venus statue. The deforming cylinder (Wobble) demonstrates that the proposed method can handle deforming objects; a sine wave is used for the deformation. For the Desk scene a chair moves along the floor. For the Bunnies scene results are presented for 1, 2, 4 and 9 bouncing bunnies, illustrating how the algorithm scales as the number of dynamic objects increases. For the Conference scene a green bunny moves across the table. Finally, for Sibenik we move a yellow rectangular object within the scene.

Table 3 shows the results, in frames per second, for rendering animations with Whitted ray tracing (RT), IGI, static instant caching (IC - without temporal coherence, the cache is cleared after each frame), and temporal instant caching (TIC). These results show that temporal instant caching is between 2.42 and 6.82 times faster than IGI. Furthermore, the results show how temporal instant caching is faster than static instant caching. The performance of temporal instant caching decreases when there is not much temporal coherence to maintain, such as in the case of the multiple bouncing bunnies. The IGI and static instant cache remain constant as the number of dynamic objects increases. As expected, the performance of the temporal instant caching reduces to that of the static instant cache as this number increases (compare Bunnies 1 with Bunnies 9).

Animations	**IGI**	**IC**	**TIC**	S_{IGI}	S_{IC}
Cornell Box	0.43	0.09	0.03	14.33	3.00
Wobble	0.37	0.09	0.03	12.33	3.00
Desk	0.56	0.09	0.03	18.67	3.00
Bunnies 1	0.56	0.12	0.03	18.67	4.00
Bunnies 2	0.55	0.13	0.04	13.75	3.25
Bunnies 4	0.55	0.13	0.05	11.00	2.60
Bunnies 9	0.57	0.12	0.09	6.33	1.33
Conference	1.23	0.17	0.05	24.60	3.40
Sibenik	0.97	0.42	0.05	19.40	8.40

Table 4: *Rendering times, in seconds, averaged over 100 frames, for rendering the diffuse interreflections only. Speedup is shown for TIC vs. IGI (S_{IGI}) and TIC vs. IC (S_{IC}).*

Since the proposed caching technique aims at speeding-up diffuse interreflections, Table 4 demonstrates the timings, in seconds, for this lighting component only, for IGI, IC and TIC. The comparison between static and temporal instant caching clearly shows the efficiency of temporal coherence exploitation, with speedups of 1.33 to 8.4 times faster than instant caching. The worst speedup is obtained with Bunnies 9, where given the large number of dynamic objects there is not much temporal coherence to be exploited. The differences in speedup relatively to the values presented in Table 3 are due to constant costs, such as primary rays, direct illumination and image display.

4.3. Validation

In order to validate our results we compared temporal instant caching with IGI and a path traced image (shot with 2,500 samples per pixel) using the perceptual metric HDR-VDP [MDMS05]. We selected the 100th frame for the comparisons of the animated scenes. HDR-VDP identifies the perceptual differences between two HDR images by taking into account limitations of the human visual system. HDR-VDP generates a certainty map detailing for each pixel the probability that a human observer will detect a difference. This may be visualised as an image that highlights the differences between the two images in false colour, with the areas most likely to be considered perceptually different in red, those less noticeable in green and those not considered perceptually different at all in grey. Figure 5 visualises the

(a) Path traced (b) VDP TIC vs. PT (c) TIC

(d) IGI (e) VDP TIC vs. IGI

Figure 5: *HDR-VDP comparisons for the Office scene. Areas most likely to be considered perceptually different in red, those less noticeable in green and those considered perceptually equivalent in grey. Note that many errors appear around blinds, which are primarily caused due to the IGI and TIC using only one sample per pixel. The path traced image has many more samples per pixel (*$2,500$ *spp) which removes aliasing artifacts.*

VDP Results	**P(X) > 75%**		**P(X) > 95%**	
	PT	**IGI**	**PT**	**IGI**
Cornell Box	0.57	0.00	0.31	0.00
Wobble	0.13	0.00	0.05	0.00
Desk	2.11	0.02	1.01	0.01
Bunnies 9	0.21	0.00	0.06	0.00
Cornell specular	0.91	0.00	0.43	0.00
Cornell Indirect	1.86	0.33	0.88	0.19
Conference	1.61	0.16	0.91	0.06
Sibenik	0.18	0.06	0.03	0.00
Office	0.45	0.23	0.00	0.00

Table 5: *Results of the HDR-VDP comparison between temporal instant caching and path tracing (PT), and temporal instant caching and IGI. Numbers indicate the % of pixels that might be perceived as different by a human observer with a probability larger than 75% or 95%, respectively.*

HDR-VDP results in false colour for the Office scene. Furthermore, the results can be summarised in a single numeric value that gives the percentage of pixels out of the entire image that are likely to be perceived as different within a given probability. We present results for $P(X) \geq 0.75$ and $P(X) \geq 0.95$, which indicate the percentage of pixels that are likely to be detected with a probability greater than or equal to 0.75 and 0.95 respectively. Table 5 shows our results when comparing temporal instant caching with a path traced image and with IGI. When comparing with IGI our results showed no more than a 0.33% difference. This shows that although instant caching results in an improvement in performance there is no perceived loss in quality when compared to IGI. Furthermore, as can be seen in the results, there is very little perceptual difference when compared to path traced images, that require several minutes of computation for each frame on modern multicore machines, with no more than 2.11% difference. It also must be noted that the temporal instant caching and IGI shoot only one ray per pixel while the path tracing's 2,500 samples per pixel reduce aliasing considerably and will account, in part, for the perceptual error detected, see Figure 5.

5. Discussion and Future Work

One of the limitations of methods based on instant radiosity is that their rendering time is, for the most part, linearly dependent on the number of VPLs that are shot. For the case of the temporal instant cache, the situation is further aggravated since the temporal update is also dependent on the number of VPLs. Several approaches have been proposed to improve instant radiosity-based algorithms and alleviate this problem; these will be explored in future work. Importance has been used in the past to direct VPL placement for complex environments [WBS03], thus reducing the number of VPLs that are required to be shot. Temporal-awareness was used to ensure that coherence between VPL placement was maintained. This importance with temporal-

awareness would allow the temporal instant cache to operate without large numbers of VPLs for complex highly occluded scenes. This would be integrated into temporal instant cache as a preprocess before shooting the VPLs, requiring little changes to the temporal instant cache method as presented here, and few modifications to the algorithm presented by Wald et al. [WBS03]. Other similar work [SIMP06, SIP07], will be investigated in the future, to help improve VPL distribution.

Lightcuts [WFA*05, WABG06] have been used to cluster point light sources (or VPLs) and reduce the VPL processing count. Their utilisation in conjunction with the temporal instant cache can be mutually beneficial, and will be investigated in the future. This may require adding some form of temporal criteria to lightcuts to ensure that the VPL clustering does not change drastically over frames. Lightcuts require building the clusters binary tree and selecting, for each shading point, the most appropriate cut. These operations incur non-negligible overheads that might compromise interactivity; careful optimisation and eventual relaxation of the clustering and cut selection criteria might be required.

One aspect common to most caching mechanisms is that the search for cached samples to interpolate from can impact the performance as the cache count increases. This problem is further accentuated in our method since the cache count effects the update computation. Since samples computed in earlier frames might not be re-used (if they do not contribute to the current view point), ageing methods similar to those presented by Tawara et al. [TMS04], whereby cached samples that are infrequently used are discarded, would directly improve the performance of our method. The test for ageing would not impact on the current temporal instant cache method and could be integrated as part of the update function when cycling through each cached sample (Line 2, Algorithm 1).

Our algorithm supports on-demand reevaluation of the visibility rays, which although partially improving performance still requires major parts of the computation for each sample to be executed each frame. An alternative approach would be to only update cached samples when requested for interpolation, which would entail that the update is performed on demand. This would require maintaining a structure consisting of those VPLs which are invalidated and those objects that would have moved at each frame. The on-demand cached sample update would need to query this structure to identify which visibility rays to update. Ageing would benefit this method by placing an upperbound on the number of frames that the structure would need to store. Due to the non-trivial additions to temporal instant caching, we propose to investigate this in future work. Future work will also look into computing gradients appropriate for the instant cache, and possibly the temporal method also, which, as with similar gradient methods [WH92], could reduce the number of cached samples and help mitigate this issue.

6. Conclusion

In this paper we have presented a caching scheme for accelerating ray tracing with indirect diffuse interreflections to interactive rates for dynamic scenes. The design of our algorithm has made it straightforward to extend spatial coherence into the temporal domain. The temporal coherence used for temporal instant caching makes it possible to avoid computation that is wasteful by computing only what is needed. The use of spatial and temporal coherence allows us to maintain high frame rates; however, when little temporal coherence remains, the spatial coherence is typically enough to maintain reasonable frame rates which are competitive with some of the best CPU methods such as IGI.

7. Acknowledgement

We thank Greg Ward for the Office and Conference scenes from Radiance package, Marko Dabrovic for the Šibenik cathedral model. Finally, we thank Stanford's Graphics Group for the Bunny model from the Stanford 3D Repository. We would like to thank Kadi Bouatouch for insightful comments on the document. This research was partially funded PT-FCT grant PTDC/EIA/65965/2006 (IGIDE project) and UK-EPSRC grant EP/D069874/2.

References

[AFO05] ARIKAN O., FORSYTH D. A., O'BRIEN J. F.: Fast and detailed approximate global illumination by irradiance decomposition. In *Proceedings of ACM SIGGRAPH 2005* (2005), ACM Press.

[BWG03] BALA K., WALTER B., GREENBERG D. P.: Combining edges and points for interactive high-quality rendering. *ACM Trans. Graph. 22*, 3 (2003), 631–640.

[DSDD07] DACHSBACHER C., STAMMINGER M., DRETTAKIS G., DURAND F.: Implicit visibility and antiradiance for interactive global illumination. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers* (New York, NY, USA, 2007), ACM, p. 61.

[DWWL05] DAYAL A., WOOLLEY C., WATSON B., LUEBKE D. P.: Adaptive frameless rendering. In *Rendering Techniques* (2005), pp. 265–275.

[GBP07] GAUTRON P., BOUATOUCH K., PATTANAIK S.: Temporal radiance caching. *IEEE Transactions on Visualization and Computer Graphics 13*, 5 (2007), 891–901.

[GKBP05] GAUTRON P., KŘIVÁNEK J., BOUATOUCH K., PATTANAIK S.: Radiance cache splatting: A GPU-friendly global illumination algorithm. In *Proceedings of Eurographics Symposium on Rendering* (June 2005).

[GTGB84] GORAL C. M., TORRANCE K. E., GREENBERG D. P., BATTAILE B.: Modeling the interaction of light between diffuse surfaces. In *SIGGRAPH '84* (1984), ACM Press, pp. 213–222.

[IDYN07] IWASAKI K., DOBASHI Y., YOSHIMOTO F., NISHITA T.: Precomputed radiance transfer for dynamics scene taking into account light interreflection. In *Eurographics Symposium on Rendering 2007* (2007), Eurographics.

[JDZJ08] JAROSZ W., DONNER C., ZWICKER M., JENSEN H. W.: Radiance caching for participating media. *ACM Trans. Graph. 27*, 1 (2008), 1–11.

[Jen01] JENSEN H. W.: *Realistic Image Synthesis Using Photon Mapping*. AK Peters, 2001.

[KBPv06] KŘIVÁNEK J., BOUATOUCH K., PATTANAIK S. N., ŽÁRA J.: Making radiance and irradiance caching practical: Adaptive caching and neighbor clamping. In *Rendering Techniques 2006, Eurographics Symposium on Rendering* (June 2006).

[Kel97] KELLER A.: Instant radiosity. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1997), ACM Press/Addison-Wesley Publishing Co., pp. 49–56.

[KGPB05] KŘIVÁNEK J., GAUTRON P., PATTANAIK S., BOUATOUCH K.: Radiance caching for efficient global illumination computation. *IEEE Transactions on Visualization and Computer Graphics 11*, 5 (2005), 550–561.

[KLC06] KENG S.-L., LEE W.-Y., CHUANG J.-H.: An efficient caching-based rendering of translucent materials. *Vis. Comput. 23*, 1 (2006), 59–69.

[LSK*07] LAINE S., SARANSAARI H., KONTKANEN J., LEHTINEN J., AILA T.: Incremental instant radiosity for real-time indirect illumination. In *Proceedings of Eurographics Symposium on Rendering 2007* (2007), Eurographics Association, pp. xx–yy.

[MDMS05] MANTIUK R., DALY S., MYSZKOWSKI K., SEIDEL H.-P.: Predicting Visible Differences in High Dynamic Range Images - Model and its Calibration. *Human Vision and Electronic Imaging X, IS&T/SPIE's 17th Annual Symposium on Electronic Imaging* (2005), 204–214.

[PH04] PHARR M., HUMPHREYS G.: *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[PMS*99] PARKER S., MARTIN W., SLOAN P.-P. J., SHIRLEY P., SMITS B., HANSEN C.: Interactive Ray Tracing. In *1999 Symposium Interactive 3D Computer Graphics* (1999), pp. 119–126.

[PWL*07] PAN M., WANG R., LIU X., PENG Q., BAO H.: Precomputed radiance transfer field for rendering interreSSections in dynamic scenes. *Computer Graphics Forum 26*, 3 (2007).

[RGK*08] RITSCHEL T., GROSCH T., KIM M. H., SEIDEL H.-P., DACHSBACHER C., KAUTZ J.: Imperfect shadow maps for efficient computation of indirect illumination. *ACM Trans. Graph. 27*, 5 (2008), 1–8.

[SIMP06] SEGOVIA B., IEHL J.-C., MITANCHEY R., PÉROCHE B.: Bidirectional instant radiosity. In *Proceedings of the 17th Eurographics Workshop on Rendering* (2006).

[SIP07] SEGOVIA B., IEHL J.-C., PÉROCHE B.: Metropolis Instant Radiosity. *Computer Graphics Forum 26*, 3 (Sept. 2007), 425–434.

[SKDM05] SMKY M., KINUWAKI S.-I., DURIKOVIC R., MYSZKOWSKI K.: Temporally coherent irradiance caching for high quality animation rendering. In *The European Association for Computer Graphics 26th Annual Conference EUROGRAPHICS 2005* (2005), vol. 24, Blackwell.

[SKS02] SLOAN P.-P., KAUTZ J., SNYDER J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 527–536.

[TL04] TABELLION E., LAMORLETTE A.: An approximate global illumination system for computer generated films. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), ACM, pp. 469–476.

[TMS04] TAWARA T., MYSZKOWSKI K., SEIDEL H.-P.: Exploiting temporal coherence in final gathering for dynamic scenes. *cgi 00* (2004), 110–119.

[TPWG02] TOLE P., PELLACINI F., WALTER B., GREENBERG D.: Interactive Global Illumination. In *SIGGRAPH'02* (2002), ACM Press.

[WABG06] WALTER B., ARBREE A., BALA K., GREENBERG D. P.: Multidimensional lightcuts. *ACM Trans. Graph. 25*, 3 (2006), 1081–1088.

[War91] WARD G.: Real pixels. *Graphics Gems 2* (1991), 15–31.

[WBS03] WALD I., BENTHIN C., SLUSALLEK P.: Interactive Global Illumination in Complex and Highly Occluded Environments. In *Proceedings of the 14th Eurographics Workshop on Rendering* (2003).

[WBS07] WALD I., BOULOS S., SHIRLEY P.: Ray tracing deformable scenes using dynamic bounding volume hierarchies. *ACM Trans. Graph. 26*, 1 (2007), 6.

[WDP99] WALTER B., DETTRAKIS G., PARKER S.: Interactive Rendering using the Render Cache. In *Tenth Eurographics Workshop on Rendering* (1999), pp. 311–320.

[WFA*05] WALTER B., FERNANDEZ S., ARBREE A., BALA K., DONIKIAN M., GREENBERG D. P.: Lightcuts: a scalable approach to illumination. *ACM Trans. Graph. 24*, 3 (2005), 1098–1107.

[WH92] WARD G., HECKBERT P.: Irradiance Gradients. In *3rd Annual Eurographics Workshop on Rendering* (Bristol, UK, 1992).

[WKB*02] WALD I., KOLLIG T., BENTHIN C., KELLER A., SLUSALLEK P.: Interactive Global Illumination using Fast Ray Tracing. In *13th EUROGRAPHICS Workshop on Rendering* (Pisa, Italy, 2002).

[WMG*07] WALD I., MARK W. R., GÜNTHER J., BOULOS S., IZE T., HUNT W., PARKER S. G., SHIRLEY P.: State of the art in ray tracing animated scenes. In *STAR Proceedings of Eurographics 2007* (Sept. 2007), Schmalstieg D., Bittner J., (Eds.), The Eurographics Association, pp. 89–116.

[WRC88] WARD G. J., RUBINSTEIN F. M., CLEAR R. D.: A ray tracing solution for diffuse interreflection. In *SIGGRAPH '88* (1988), ACM Press, pp. 85–92.

[WSBW01] WALD I., SLUSALLEK P., BENTHIN C., WAGNER M.: Interactive Rendering With Coherent Raytracing. In *EUROGRAPHICS 2001* (Manchester, United Kingdom, September 2001), pp. 153–164.

[ZHL*05] ZHOU K., HU Y., LIN S., GUO B., SHUM H.-Y.: Precomputed shadow fields for dynamic scenes. *ACM Trans. Graph. 24*, 3 (2005), 1196–1201.